

**APPLICATION FOR
UNITED STATES PATENT
IN THE NAME OF**

JOSEPH F. CIHULA

Assigned to

INTEL CORPORATION

for

**SYSTEM FOR INCREASING REALIZED SECURE SOCKETS LAYER
ENCRYPTION AND DECRYPTION CONNECTIONS**

Prepared by:

Pillsbury Winthrop LLP
725 South Figueroa Street, Suite 2800
Los Angeles, California 90017-5406
Tel: (213) 488-7100
Fax: (213) 629-1033

ATTORNEY DOCKET NO. 081674-0280181

Express Mail No. EL 724 027 700 US

081674-0280181

SYSTEM FOR INCREASING REALIZED SECURE SOCKETS LAYER ENCRYPTION AND DECRYPTION CONNECTIONS

BACKGROUND

1. Field of the Invention

Embodiments described herein are directed to a system for increasing realized secure sockets layer (“SSL”) encryption and decryption connections without significant impact to client response. The system combines monitoring of server load with dynamic adjustment of static SSL parameters to optimize an entire system of devices.

2. Related Art

Secure Sockets Layer (“SSL”) is a protocol for transmitting private documents in a public data communication network. SSL operates by using a key to encrypt data that is transferred over an SSL connection. The SSL protocol typically uses Transmission Control Protocol/Internet Protocol (“TCP/IP”) and allows the following: 1) an SSL-enabled server to authenticate itself to an SSL-enabled client; 2) the client to authenticate itself to the server; and 3) both machines to establish an encrypted connection. An encrypted SSL connection requires the encryption by the sending software, and the decryption by the receiving software, of all information sent between a server and a client, thereby providing a high degree of confidentiality. Confidentiality is important for both parties to any private transaction. In addition, all data sent over an encrypted SSL connection is protected with a mechanism for detecting tampering—that is, for automatically determining whether the data has been altered from the point of transmission from the sending software until the data is received by the receiving software.

In current systems, SSL encryption and decryption devices (“SSL devices”) operate independently of the servers upon which they are performing the SSL operations. That is, they

do not use information from the servers to determine loading of the device that is performing the SSL encryption and decryption. Load refers to the amount of data, i.e., traffic that the device carries. Parameters for configuring the SSL performance are static, if existent at all. When thresholds for the number of connections that an SSL device will accept are available, they are static because they are the only information available to the device. Without information about server loading, the SSL device cannot make dynamic choices or decisions. The SSL device can, for example, determine when it can no longer sustain more connections. The SSL device is unable, however, to determine which server can sustain the same. The result is that overall SSL performance of a system of servers and SSL devices, with the constraint of no significant client impact, is limited by the performance of the SSL device. This is because static algorithms that determine SSL offload to individual servers cannot meet a no significant-impact guarantee.

A system for increasing realized SSL encryption and decryption connections is thus designed to combine monitoring of server load with adjustment of SSL parameters to optimize the system of devices. The result of this dynamic system is increased SSL performance without significant impact to end-user response.

BRIEF DESCRIPTION OF THE DRAWINGS

A detailed description of embodiments of the invention will be made with reference to the accompanying drawings, wherein like numerals designate corresponding parts in the several figures.

FIG. 1 is a depiction of network connectivity of one SSL device and three servers.

FIG. 2 is a flowchart illustrating the optimization process of a system of one SSL device and three servers.

DETAILED DESCRIPTION

The following paragraphs describe a system for increasing realized secure sockets layer (“SSL”) encryption and decryption connections. An embodiment of the present invention includes an SSL encryption and decryption device (“SSL device”) that includes both hardware and software. The software contains the code that performs calculations and acts on the calculations.

According to one embodiment of the present invention, as illustrated in Figure 1, a system **160a** includes one SSL device **120** located within a data communication network **110** between a set of three servers **140a-c** and a client machine **130**. Coupled to one side of the data communication network **110** is the client machine **130**. Meanwhile, a switch **135** acts as an intermediary between the SSL device **120** and the servers **140a-c**, whereby the switch **135** receives data from the SSL device **120** and then forwards the data to the servers **140a-c**.

This system **160a** is dynamically optimized within the data communication network **110**. The data communication network **110** may include the Internet, an Intranet, or any combination of public and private data communication networks. The data communication network **110** may be configured as a local-area network, wide-area network, or another kind of architecture. A multitude of systems, as depicted by **160b-c**, may further be sustained within the data communication network **110**.

The client machine **130** attempts to open SSL connections to the servers **140a-c**. The SSL device **120** intercepts these connections, performs SSL encryption and decryption, and then sends the encrypted information to the appropriate server **140a-c** in an unencrypted format. For example, if client machine **130** attempts to open an SSL connection to server **140a**, the SSL device **120** intercepts the connection and opens it with client machine **130**. The SSL device **120**

then unencrypts the data sent over the connection and sends that data to server **140a**, unencrypted.

The system combines the monitoring of server **140a-c** load with dynamic adjustment of SSL device **120** parameters to optimize the entire system. The system, i.e. software running on some platform such as, but not limited to, an SSL device or a server, monitors certain parameters of the servers **140a-c** such as, but not limited to, CPU utilization and available memory, that are known to affect the ability of the servers to process SSL connections.

The servers **140a-c** may be monitored by many mechanisms. An agent, i.e., software, may be installed on the servers **140a-c** that then communicates to the SSL device **120**. Windows NT has a protocol for remote monitoring of many types of server statistics, including CPU usage. UNIX operating systems support the remote execution of programs that can provide this information. In addition, Simple Network Management Protocol ("SNMP") may also be used for monitoring.

As shown in Figure 2, an "SSL capacity" value for each server **140a-c** is calculated and represents the capacity of that server **140a-c** to process SSL connections. This is illustrated in step **210** for server **140a**, step **220** for server **140b**, and step **230** for server **140c**. The calculation may be a direct value or a computation of values. Various algorithms may be used to determine such a value. One such algorithm is $\text{capacity} = \max [(\# \text{ processors} \times \text{processor speed in MHz}/100) \times (0.7 - \text{CPU utilization}), 0]$. Since SSL acceleration hardware may be present in some systems, another possible algorithm is $\text{capacity} = \max [(\# \text{ processors} \times \text{processor speed in MHz}/100) \times (0.7 - \text{CPU utilization}) + f_n(x), 0]$, where $f_n(x)$ represents the SSL acceleration capabilities of the SSL acceleration hardware.

The greater the SSL capacity of the server **140a-c**, the fewer SSL connections the SSL device **120** should process. The number of SSL connections processed by the SSL device **120** also depends on the load of the SSL device **120**. Load is a direct value or computation such as, but not limited to, CPU utilization. Calculating the load of the SSL device **120** is shown in step **240**. If an SSL device **120** is lightly loaded, it processes more SSL connections for all the servers **140a-c** than if it is heavily loaded.

As illustrated in step **250**, the SSL capacity value is then used to calculate an “SSL connection threshold” for that server **140a-c**. This is applied to the SSL device **120** to determine how many SSL connections the SSL device **120** should process for that server **140a-c**, as shown in step **260**. One algorithm for this calculation is $\text{threshold} = 10 \times \text{server capacity} \times \text{device CPU utilization}$. This represents the number of SSL connections that the SSL device **120** would allow to be processed by a given server **140a-c**.

Since the connection threshold for the SSL device **120** is a function of both the load of the SSL device **120** and the SSL capacity of each server **140a-c**, and these values are dynamic, the connection threshold values are recalculated periodically. The recalculation is based either on time or on additional thresholds that are functions of the SSL capacity and/or SSL device load. The result of this dynamic system is increased SSL performance without significant impact to client response.

While the above description refers to particular embodiments of the present invention, it will be understood to those of ordinary skill in the art that modifications may be made without departing from the spirit thereof. The accompanying claims are intended to cover any such modifications as would fall within the true scope and spirit of the present invention.

The presently disclosed embodiments are therefore to be considered in all respects as illustrative and not restrictive; the scope of the invention being indicated by the appended claims, rather than the foregoing description. All changes that come within the meaning and range of equivalency of the claims are therefore intended to be embraced therein.